第3章 プログラミング支援機能が充実！クラウド基盤「Fixstars Amplify」でトライ②

ナンプレを解く！制約条件の考え方を習得

広田 望

Fixstars Amplifyを使ったソフトウェア開発の理解を深めるために，少しだけ難度を上げて「ナンプレ」を解くプログラミングに挑戦してみましょう．今回は，数の分割問題にはなかった制約条件を使います．

ナンプレは縦横9×9マスに1～9までの数字を当てはめていくパズル・ゲームです．数字の入れ方には，縦，横，3×3の各ブロックに1～9の数字が各1つずつ入るというルールがあります．このパズル・ゲームを組み合わせ最適化問題として捉え，イジング・マシンで答えを計算してみましょう．制約条件の理解や定式化の練習になります．

コードの全体像は**リスト1**（次頁）に示します．上から順番に解説します．

ステップ1：解くべき問題の設定

● ナンプレの問題を定義（リスト1の①）

モジュールの読み込みは割愛して，最初は解くべき問題（ナンプレ）の設定です．**リスト1**の①は計算するナンプレの例題を設定しています．ナンプレは空欄を埋める問題ですが，データ操作としては数値の方が便利なので空欄の代わりに0を使い，2重配列で9×9マスを表しています．

● 見やすい形式に整形して出力（リスト1の②）

そのままでは問題も計算結果も見づらいため，この形式の数列を整形出力する関数print_sudokuも用意しています．3×3マスの区切りも見やすいように，線を入れています（**図1**）．これらのコードは基本的なPythonのプログラミングなので，詳細な解説は割愛します．

解くべき問題の初期値が用意できたところで，実際に計算するコーディングに入っていきましょう．まずは定式化からです．

ステップ2：多重配列で決定変数を用意する

● 定式化の戦略…ワンホット・エンコーディング

定式化は解くべき問題を数式で表すことです．今回はナンプレを解くので，ナンプレのルールを数式で表すことを考えます．

数式でナンプレのルールを表す戦略を考えてみましょう．最適化する組み合わせは，各マスにどの値が入るかというものです．81個の各マスには1～9までの数字が入るので，単純計算で組み合わせのパターン数は9^{81}となります．これは無量大数の約20億倍にもおよびます．

もし決定変数が9通り以上の値を取れれば，81個の変数を用意するだけでナンプレの組み合わせを表現できるでしょう．しかし，イジング・マシンを使って問題を解く場合，変数は2通りの値しか取れません．2通りの値が取れる変数で9通りの値を表現する戦略は幾つかありますが，ここでは分かりやすくワンホット・エンコーディングを使いましょう．

まず，各マスごとに変数を9個用意します．変数は0か1だけを取るバイナリ変数を使います．このとき，9個の変数の中で1を取れるのは1個だけに制限します．1つだけ値をもつワンホットな9個の変数配列によって，1～9のどの値なのかを表現するのです（**図2**）．

```
7 0 0 | 0 1 0 | 0 0 4
5 2 0 | 0 0 0 | 0 8 0
0 0 0 | 0 0 0 | 0 0 0   ← 0は空欄を意味する
---------------------
0 0 9 | 0 0 4 | 0 0 3
3 8 0 | 6 5 0 | 0 0 0
0 5 0 | 0 2 0 | 0 0 0
---------------------
0 0 0 | 0 0 0 | 3 0 0
4 0 7 | 0 0 0 | 2 0 0
0 0 8 | 0 0 7 | 0 9 0
```

図1　定義したナンプレの問題を見やすい形式で出力した結果
リスト1における②の出力結果．0の部分は空欄を意味する